Optimization Project - Kyoto Restaurants, models and maps in R

Alisa Babikova, WanQi (Angie) Tay, Yingkun (Will) Zhu

August 21, 2018

```
library(ompr)
library(dplyr)
library(ROI)
library(ROI.plugin.glpk)
library(ompr.roi)
```

```
#Load the dataset that has been cleaned up in OpenRefine
data = read.csv("Kyoto.csv")
head(data)
```

```
# For optimization models we will need 6 variables out of the original dataframe
kyoto = data[, c(4,13,18,21,22,26,24,25)]
kyoto[,2] = round(kyoto[,2], 2)
kyoto[,3] = round(kyoto[,3], 2)
kyoto[is.na(kyoto)] = 0
head(kyoto)
```

```
##                           Name AverageDinnerPriceInUSD
## 1         Orudeidainingurajou                     40.45
## 2 Steak Frites Gaspard zinzin                     31.46
## 3                      KAZUMA                     31.46
## 4   okonomiyakiteppanyakimiki                     31.46
## 5                      Sumika                     31.46
## 6                    Gayagaya                     31.46
##   AverageLunchPriceInUSD DinnerRating LunchRating    Category      Lat
## 1                  22.47         3.20        3.38      Buffet 34.98618
## 2                  13.48         3.06        3.33      Bistro 35.00583
## 3                   0.00         3.28        0.00       Other 35.00658
## 4                   0.00         3.14        0.00 Okonomiyaki 34.99592
## 5                   0.00         3.34        0.00      Fusion 35.00470
## 6                   0.00         3.04        0.00         Bar 35.00717
##       Long
## 1 135.7613
## 2 135.7599
## 3 135.7702
## 4 135.7483
## 5 135.7702
## 6 135.7700
```

First Model - Dinner [USING OMPR - MIP Model]

- the following three models will be using the parameters demonstrated in the presentation

```
preference = c("Okonomiyaki", "Sukiyaki", "Bar", "Horumon", "Sumibiyaki", "Tonkatsu", "Mizutaki", "Kushiyaki")
days = 5
budget = c(200, 500)
df = kyoto[kyoto[, 6] %in% (preference), ] # Filter out preferences that are not listed above
n = nrow(df)
p = length(preference)

dModel = MIPModel() %>%
          add_variable(x[i], i = 1:n, type = "binary") %>%
          set_objective(sum_expr(df[i, 4] * x[i], i = 1:n)/days, sense = "max") %>% # rating
          add_constraint(sum_expr(df[i, 2] * x[i], i = 1:n) <= max(budget)) %>% # dinner maximumn budget
          add_constraint(sum_expr(df[i, 2] * x[i], i = 1:n) >= min(budget)) %>% # dinner minimum budget
          add_constraint(sum_expr(x[i], i = 1:n) == days)

for (j in 1:p) {
          add_constraint(dModel, sum_expr(as.numeric(df[i, 6] == preference[j]) * x[i], i = 1:n) <= 1)
}

(d.Obj.value = solve_model(dModel, with_ROI("glpk")))
```

```
## Status: optimal
## Objective value: 3.414
```

```
dOpt = get_solution(d.Obj.value, x[i]) %>%
  dplyr::filter(value > 0)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```

```
dOpt
```

```
##   variable  i value
## 1        x 10     1
## 2        x 19     1
## 3        x 34     1
## 4        x 40     1
## 5        x 41     1
```

```
(dModel_solution = df[dOpt$i,c(1,2,4,6)])
```

```
##                              Name AverageDinnerPriceInUSD DinnerRating
## 107        torisemmontenyamadori                    31.46         3.38
## 218 kaitoshirowainnobarukakimaru                    31.46         3.31
## 366                     L'ajitto                    80.91         3.23
## 531                       Yumeya                    13.48         3.55
## 566                   Takasegawa                    49.44         3.60
##        Category
## 107    Sukiyaki
## 218         Bar
## 366 Okonomiyaki
## 531 Okonomiyaki
## 566  Sumibiyaki
```

```
mean(dModel_solution[,3])
```

```
## [1] 3.414
```

```
sum(dModel_solution[,2])
```

```
## [1] 206.75
```

Dinner Model:

Objective value: 3.414 [Average Rating]

Total Dinner Price: $ 206.75

Second Model - Lunch [USING OMPR - MIP Model]

```
preference = c("Buffet", "Bistro", "Fusion", "Bar", "Yakitori")
days = 2
budget = c(50, 80)
df = kyoto[kyoto[, 6] %in% (preference), ]
n = nrow(df)
p = length(preference)

lModel = MIPModel() %>%
          add_variable(x[i], i = 1:n, type = "binary") %>%
          set_objective(sum_expr(df[i, 5] * x[i], i = 1:n)/days, sense = "max") %>% # rating
          add_constraint(sum_expr(df[i, 3] * x[i], i = 1:n) <= max(budget)) %>% # Lunch price
          add_constraint(sum_expr(df[i, 3] * x[i], i = 1:n) >= min(budget)) %>%
          add_constraint(sum_expr(x[i], i = 1:n) == days)

for (j in 1:p) {
          add_constraint(lModel, sum_expr(as.numeric(df[i, 6] == preference[j]) * x[i], i = 1:n) <= 1)

  }
(l.Obj.value = solve_model(lModel, with_ROI("glpk")))
```

```
## Status: optimal
## Objective value: 3.695
```

```
lOpt = get_solution(l.Obj.value, x[i]) %>%
  dplyr::filter(value > 0)

lOpt
```

```
##   variable   i value
## 1        x  74     1
## 2        x 101     1
```

```
(lModel_solution = df[lOpt$i,c(1,3,5,6)])
```

```
##                 Name AverageLunchPriceInUSD LunchRating Category
## 410 Kyounozenkuruma                   13.48        3.54 Yakitori
## 609            ORTO                   40.45        3.85   Fusion
```

```
mean(lModel_solution[,3])
```

```
## [1] 3.695
```

```
sum(lModel_solution[,2])
```

```
## [1] 53.93
```

Lunch Model:

Objective value: 3.695 [Average Rating]

Total Lunch Price: $ 53.93

Third Model - Combined Model [USING OMPR - MIP Model]

```
preference = c("Okonomiyaki", "Sukiyaki", "Buffet", "Ramen", "Sumibiyaki", "Tonkatsu", "Mizutaki")
days = 3
budget = c(120, 180)
df = kyoto[kyoto[, 6] %in% (preference), ]
n = nrow(df)
p = length(preference)

cModel = MIPModel() %>%
          add_variable(x[i], i = 1:n, type = "binary") %>% # x = dinner
          add_variable(y[i], i = 1:n, type = "binary") %>% # y = lunch
          set_objective(sum_expr((df[i, 4] * x[i]) + (df[i, 5] * y[i]), i = 1:n) / (2 * days), sense = "max") %>%
          add_constraint(sum_expr((df[i, 2] * x[i]) + (df[i, 3] * y[i]), i = 1:n) <= max(budget)) %>%
          add_constraint(sum_expr((df[i, 2] * x[i]) + (df[i, 3] * y[i]), i = 1:n) >= min(budget)) %>%
          add_constraint(sum_expr(x[i], i = 1:n) == days) %>%
          add_constraint(sum_expr(y[i], i = 1:n) == days) %>%
          add_constraint((x[i] + y[i]) <= 1, i = 1:n)

for (j in 1:p) {
          add_constraint(cModel, sum_expr((as.numeric(df[i, 6] == preference[j]) * x[i]) +
                                            (as.numeric(df[i,6] == preference[j]) * y[i]), i = 1:n) <= 1)

}

(c.Obj.value = solve_model(cModel, with_ROI("glpk")))
```

```
## Status: optimal
## Objective value: 3.548333
```

```
c.dOpt = get_solution(c.Obj.value, x[i]) %>%
  dplyr::filter(value > 0)

c.lOpt = get_solution(c.Obj.value, y[i]) %>%
  dplyr::filter(value > 0)
```

```
x = cbind(df[c.dOpt$i,c(1,2,3,4,5,6)], Choice = "Dinner")
y = cbind(df[c.lOpt$i,c(1,2,3,4,5,6)], Choice = "Lunch")
(cModel_solution = rbind(x,y))
```

```
##                          Name AverageDinnerPriceInUSD
## 185         Mossannobetayaki                     22.47
## 482       chuukasobatakayasu                      4.49
## 566               Takasegawa                     49.44
## 46       o-rudeidainingukaza                     40.45
## 385 Cafe Restaurant Le Temps                     49.44
## 650         Menshoutakamatsu                      4.49
##     AverageLunchPriceInUSD DinnerRating LunchRating    Category Choice
## 185                   4.49         3.51        3.06 Okonomiyaki Dinner
## 482                   4.49         3.58        3.59       Ramen Dinner
## 566                   0.00         3.60        0.00  Sumibiyaki Dinner
## 46                   22.47         3.08        3.52      Buffet  Lunch
## 385                  22.47         3.26        3.50      Buffet  Lunch
## 650                   4.49         3.41        3.58       Ramen  Lunch
```

```
(sum(cModel_solution[,4][cModel_solution[,7] == "Dinner"]) + sum(cModel_solution[,5][cModel_solution[,7] == "Lunch"]))/nro
w(cModel_solution)
```

```
## [1] 3.548333
```

```
(sum(cModel_solution[,2][cModel_solution[,7] == "Dinner"]) + sum(cModel_solution[,3][cModel_solution[,7] == "Lunch"]))
```

```
## [1] 125.83
```

Combined Model:

Objective value: 3.548333

Total Price: $125.83

Visualization

```
my.kyoto <- read.csv("Kyoto.csv")

LatList <- c(my.kyoto$Lat)
#LatList
LonList <- c(my.kyoto$Long)
#LonList

LatLong.Kyoto <- data.frame(X = LatList, Y = LonList)
#LatLong.Kyoto

names(LatLong.Kyoto) <- c("X","Y")
```

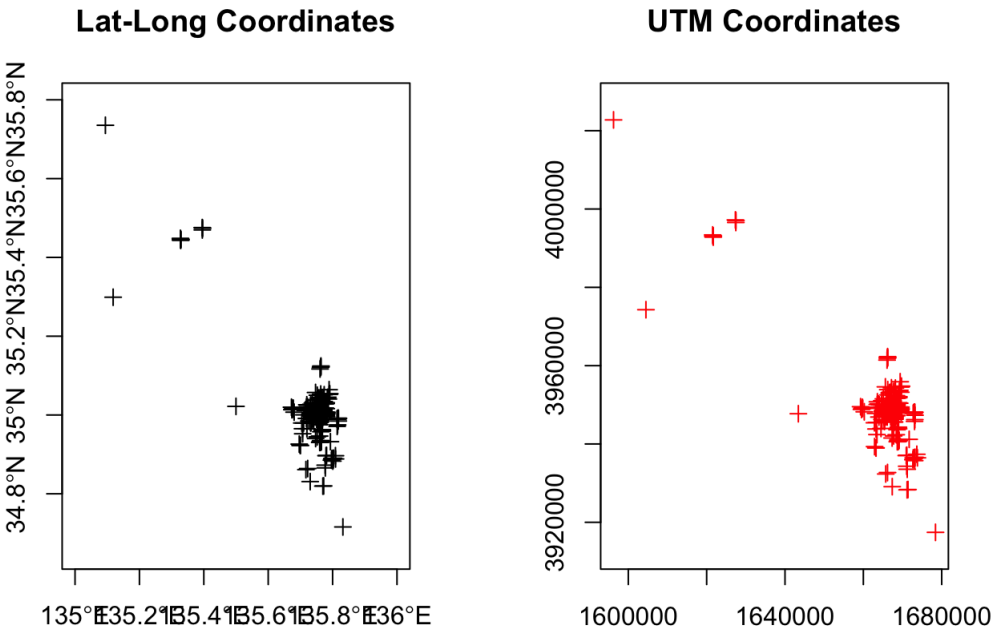
```
# Convert it to a spatial object
coordinates(LatLong.Kyoto) <- ~ Y + X # longitude first

# Add a coordinate reference system
proj4string(LatLong.Kyoto) <- CRS("+proj=longlat +ellps=WGS84 +datum=WGS84")

# Project using spTransform
UtmList <- spTransform(LatLong.Kyoto, CRS("+proj=utm +zone=51N ellps=WGS84"))
UtmList2 <- na.omit(UtmList)
summary(UtmList2)
```

```
## Object of class SpatialPoints
## Coordinates:
##       min     max
## Y 1596232 1678415
## X 3917435 4022756
## Is projected: TRUE
## proj4string :
## [+proj=utm +zone=51N ellps=WGS84 +ellps=WGS84]
## Number of points: 655
```

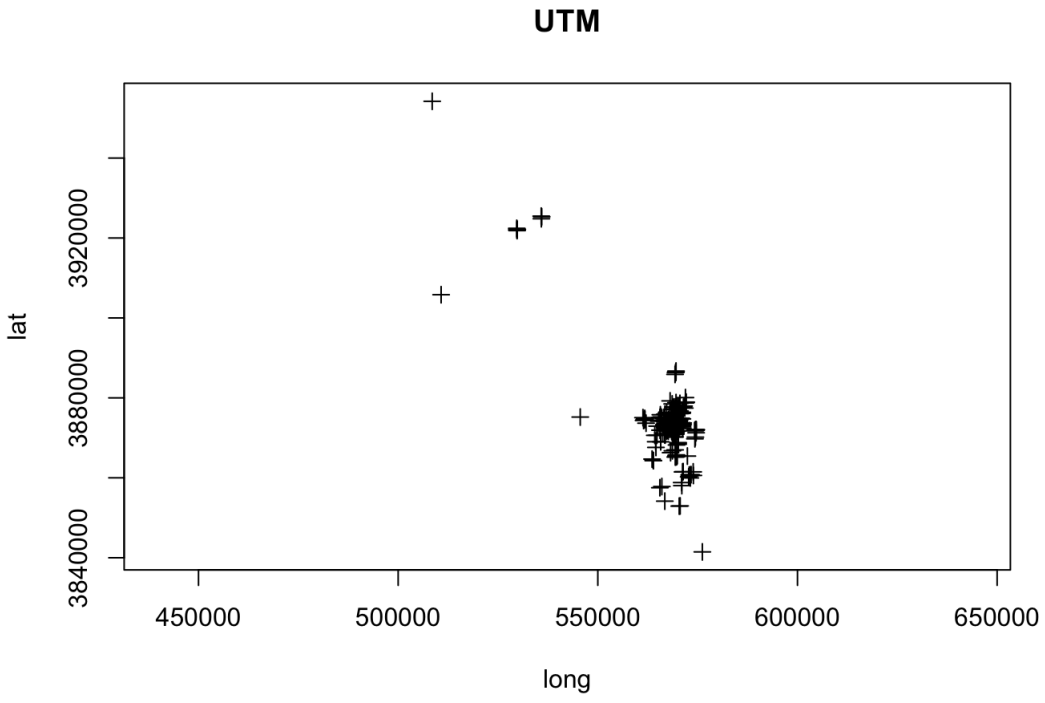
```
par(mfrow = c(1, 2))
plot(LatLong.Kyoto, axes = TRUE, main = "Lat-Long Coordinates", xlim = c(135:136))
plot(UtmList2, axes = TRUE, main = "UTM Coordinates", col = "red")
```



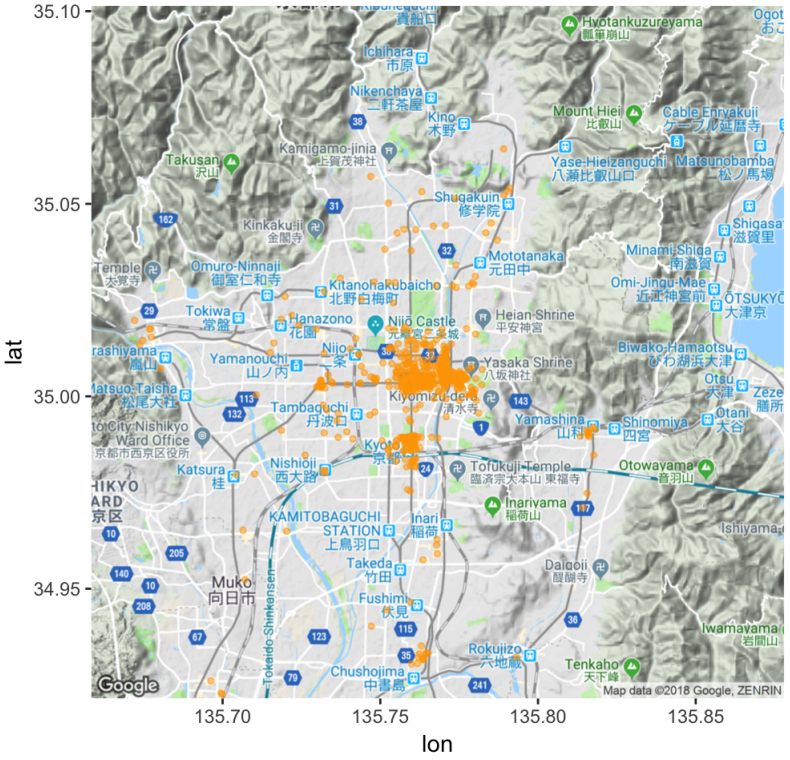

```
cord.dec <- SpatialPoints(cbind(my.kyoto$Long, my.kyoto$Lat), proj4string=CRS("+proj=longlat"))
cord.UTM <- spTransform(cord.dec, CRS("+init=epsg:3094"))
# According to Google, Japan's epsg code is 3094 or 3095
```

```
# An overview
map <- get_map('Kyoto', maptype = 'terrain', source = 'google', zoom = 12)

plot(cord.UTM, axes = TRUE, xlab = "long", ylab = "lat", main = "UTM")
```



```
ggmap(map)+
geom_point(aes(x = my.kyoto$Long, y = my.kyoto$Lat), data = as.data.frame(coordinates(cord.dec)),
         alpha = .5, color = "orange", size = 1)
```




```
# get the map as canvas
kiki <- get_map("kyoto", zoom = 13)
```

Map for Dinner selection

```
dinner.selection <- data.frame(lng = df[dOpt$i,c(7,8)][, 2], lat = df[dOpt$i,c(7,8)][, 1])

dinner <- ggmap(kiki) + geom_point(data = dinner.selection, aes(lng, lat), size = 3, col = "red", alpha = 0.5)
dinner
```

Map for Lunch Selection

```
lunch.selection <- data.frame(lng = df[lOpt$i,c(7,8)][, 2], lat = df[lOpt$i,c(7,8)][, 1])

lunch <- ggmap(kiki) + geom_point(data = lunch.selection, aes(lng, lat), size = 3, col = "red", alpha = 0.5)
lunch
```

Map for Combined Model Selection

```
d1 <- data.frame(lng=df[c.dOpt$i,c(7,8)][, 2], lat=df[c.dOpt$i,c(7,8)][, 1])
d2 <- data.frame(lng=df[c.lOpt$i,c(7,8)][, 2], lat=df[c.lOpt$i,c(7,8)][, 1])
combined.selection <- rbind(d1, d2)

combined <- ggmap(kiki) + geom_point(data = combined.selection, aes(lng, lat), size = 3, col = "red", alpha = 0.5)
combined
```

The End